

# Parameterized Approximation Schemes for Steiner Trees with Small Number of Steiner Vertices

**Pavel Dvořák**

Computer Science Institute, Charles University, Prague, Czechia  
koblich@iuuk.mff.cuni.cz

**Andreas Emil Feldmann**

Department of Applied Mathematics, Charles University, Prague, Czechia  
feldmann.a.e@gmail.com

**Dušan Knop**

Department of Informatics, University of Bergen, Bergen, Norway and  
Department of Applied Mathematics, Charles University, Prague, Czechia  
knop@kam.mff.cuni.cz

**Tomáš Masařík**

Department of Applied Mathematics, Charles University, Prague, Czechia  
masarik@kam.mff.cuni.cz

**Tomáš Toufar**

Computer Science Institute, Charles University, Prague, Czechia  
toufar@iuuk.mff.cuni.cz

**Pavel Veselý**

Computer Science Institute, Charles University, Prague, Czechia  
vesely@iuuk.mff.cuni.cz

---

## Abstract

We study the STEINER TREE problem, in which a set of *terminal* vertices needs to be connected in the cheapest possible way in an edge-weighted graph. This problem has been extensively studied from the viewpoint of approximation and also parametrization. In particular, on one hand STEINER TREE is known to be APX-hard, and  $W[2]$ -hard on the other, if parameterized by the number of non-terminals (*Steiner vertices*) in the optimum solution. In contrast to this we give an *efficient parameterized approximation scheme* (EPAS), which circumvents both hardness results. Moreover, our methods imply the existence of a *polynomial size approximate kernelization scheme* (PSAKS) for the considered parameter.

We further study the parameterized approximability of other variants of STEINER TREE, such as DIRECTED STEINER TREE and STEINER FOREST. For neither of these an EPAS is likely to exist for the studied parameter: for STEINER FOREST an easy observation shows that the problem is APX-hard, even if the input graph contains no Steiner vertices. For DIRECTED STEINER TREE we prove that computing a constant approximation for this parameter is  $W[1]$ -hard. Nevertheless, we show that an EPAS exists for UNWEIGHTED DIRECTED STEINER TREE. Also we prove that there is an EPAS and a PSAKS for STEINER FOREST if in addition to the number of Steiner vertices, the number of connected components of an optimal solution is considered to be a parameter.

**2012 ACM Subject Classification** Mathematics of computing → Combinatorics, Mathematics of computing → Graph theory, Theory of computation → Design and analysis of algorithms

**Keywords and phrases** Steiner Tree, Steiner Forest, Approximation Algorithms, Parameterized Algorithms, Lossy Kernelization

**Digital Object Identifier** 10.4230/LIPIcs.STACS.2018.26



© Pavel Dvořák, Andreas E. Feldmann, Dušan Knop, Tomáš Masařík, Tomáš Toufar, and Pavel Veselý; licensed under Creative Commons License CC-BY

35th Symposium on Theoretical Aspects of Computer Science (STACS 2018).

Editors: Rolf Niedermeier and Brigitte Vallée; Article No. 26; pp. 26:1–26:15

Leibniz International Proceedings in Informatics



Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



**Related Version** The full preprinted version of this article is available at <https://arxiv.org/abs/1710.00668v2>, [17].

**Funding** This work was partially supported by the project SVV-2017-260452. D. Knop, T. Masařík and P. Veselý were supported by project 17-09142S of GAČR. D. Knop was supported by the NFR MULTIVAL project. A.E. Feldmann was supported by the Czech Science Foundation GAČR (grant #17-10090Y). A.E. Feldmann, T. Masařík, T. Toufar, and P. Veselý were supported by the project GAUK 1514217. P. Dvořák has received funding from the European Research Council under the European Union’s Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement n. 616787.

**Acknowledgements** We would like to thank Michael Lampis and Édouard Bonnet for helpful discussions on the problem. Also we thank Jiří Sgall and Martin Böhm for finding a mistake in the proof of Lemma 9.

## 1 Introduction

In this paper we study several variants of the STEINER TREE problem. In its most basic form this optimization problem takes an undirected graph  $G = (V, E)$  with edge weights  $w(e) \in \mathbb{R}_0^+$  for every  $e \in E$ , and a set  $R \subseteq V$  of *terminals* as input. The non-terminals in  $V \setminus R$  are called *Steiner vertices*. A *Steiner tree* is a tree in the graph  $G$ , which spans all terminals in  $R$  and may contain some of the Steiner vertices. The objective is to minimize the total weight  $\sum_{e \in E(T)} w(e)$  of the computed Steiner tree  $T \subseteq G$ . This fundamental optimization problem is one of the 21 original NP-hard problems listed by Karp [25] in his seminal paper from 1972, and has been intensively studied since then. The STEINER TREE problem and its variants have applications in network design, circuit layouts, and phylogenetic tree reconstruction, among others (see survey [23]).

Two popular ways to handle the seeming intractability of NP-hard problems are to design *approximation* [35] and *parameterized* [12] algorithms. For the former, an  $\alpha$ -*approximation* is computed in polynomial time for some factor  $\alpha$  specific to the algorithm, i.e., the solution is always at most a multiplicative factor of  $\alpha$  worse than the optimum of the input instance. The STEINER TREE problem, even in its basic form as defined above, is APX-hard [11], i.e., it is NP-hard to obtain an approximation factor of  $\alpha = \frac{96}{95} \approx 1.01$ . However a factor of  $\alpha = \ln(4) + \varepsilon \approx 1.39$  can be achieved in polynomial time [5], which is the currently best factor known for this runtime.

For parameterized algorithms, an instance is given together with a *parameter*  $p$  describing some property of the input. The optimum solution is computed in time  $f(p) \cdot n^{O(1)}$ , where  $f$  is a computable function independent of the input size  $n$ . If such an algorithm exists, we call the problem *fixed-parameter tractable* (FPT) for parameter  $p$ . A well-studied parameter for the STEINER TREE problem is the number of terminals  $|R|$ . It is known since the classical result of Dreyfus and Wagner [16] that the STEINER TREE problem is FPT for this parameter, as the problem can be solved in time  $3^{|R|} \cdot n^{O(1)}$  if  $n = |V|$ . This can be improved to  $2^{|R|} \cdot n^{O(1)}$  if the input graph is unweighted using the results of Björklund et al. [2]. A somewhat complementary and less-studied parameter to the number of terminals is the number of Steiner vertices in the optimum solution, i.e.,  $p = |V(T) \setminus R|$  if  $T$  is an optimum Steiner tree. It is known [15] that STEINER TREE is W[2]-hard for parameter  $p$  and therefore is unlikely to be FPT, in contrast to the parameter  $|R|$ . This parameter  $p$  has been mainly studied in the context of unweighted graphs before. The problem remains W[2]-hard in this special case and therefore the focus has been on designing parameterized algorithms for restricted graph classes, such as planar or  $d$ -degenerate graphs [24, 33].

In contrast to this, our question is: what can be done in the most general case, in which the class of input graphs is unrestricted and edges may have weights? Our main result is that we can overcome the APX-hardness of STEINER TREE on one hand, and on the other hand also the W[2]-hardness for our parameter of choice  $p$ , by combining the two paradigms of approximation and parametrization. This relatively new and growing area has gained quite a bit of attention recently (see e.g., [3, 6, 8, 9, 10, 18, 20, 26, 27, 28, 29, 32, 34]). We show that there is an *efficient parameterized approximation scheme* (EPAS), which for any  $\varepsilon > 0$  computes a  $(1 + \varepsilon)$ -approximation in time  $f(p, \varepsilon) \cdot n^{O(1)}$  for a computable function  $f$  independent of  $n$ . Note that here we consider the approximation factor of the algorithm as a parameter as well, which accounts for the “efficiency” of the approximation scheme (analogous to an *efficient polynomial time approximation scheme* or EPTAS). In fact, as summarized in the following theorem, our algorithm computes an approximation to the cheapest tree having at most  $p$  Steiner vertices, even if better solutions with more Steiner vertices exist.

► **Theorem 1.** *There is an algorithm for STEINER TREE, which given an edge-weighted undirected graph  $G = (V, E)$ , terminal set  $R \subseteq V$ ,  $\varepsilon > 0$ , and integer  $p$ , computes a  $(1 + \varepsilon)$ -approximation to the cheapest Steiner tree  $T \subseteq G$  with  $p \geq |V(T) \setminus R|$  in time  $2^{O(p^2/\varepsilon^4)} \cdot n^{O(1)}$ .<sup>1</sup>*

Many variants of the STEINER TREE problem exist, and we explore the applicability of our techniques to some common ones. For the DIRECTED STEINER TREE problem the aim is to compute an *arborescence*, i.e., a directed graph obtained by orienting the edges of a tree so that exactly one vertex called the *root* has in-degree zero (which means that all vertices are reachable from the root). More concretely, the input consists of a directed graph  $G = (V, A)$  with arc weights  $w(a) \in \mathbb{R}_0^+$  for every  $a \in A$ , a terminal set  $R \subseteq V$ , and a specified terminal  $r \in R$ . A *Steiner arborescence* is an arborescence in  $G$  with root  $r$  containing all terminals  $R$ . The objective is to find a Steiner arborescence  $T \subseteq G$  minimizing the weight  $\sum_{a \in A(T)} w(a)$ . This problem is notoriously hard to approximate: no  $O(\log^{2-\varepsilon}(n))$ -approximation exists unless  $\text{NP} \subseteq \text{ZTIME}(n^{\text{polylog}(n)})$  [22]. But even for the UNWEIGHTED DIRECTED STEINER TREE problem in which each arc has unit weight, a fairly simple reduction from the SET COVER problem implies that no  $((1 - \varepsilon) \ln n)$ -approximation algorithm is possible unless  $\text{P} = \text{NP}$  [13, 22]. At the same time, even UNWEIGHTED DIRECTED STEINER TREE is W[2]-hard for our considered parameter  $p$  [24, 30], just as for the undirected case. For this reason, all previous results have focused on restricted inputs: Jones et al. [24] prove that when combining the parameter  $p$  with the size of the largest excluded topological minor of the input graph, UNWEIGHTED DIRECTED STEINER TREE is FPT. They also show that if the input graph is acyclic and  $d$ -degenerate, the problem is FPT for the combined parameter  $p$  and  $d$ .

Our focus again is on general unrestricted inputs. We are able to leverage our techniques to the unweighted directed setting, and obtain an EPAS, as summarized in the following theorem. Here the cost of a Steiner arborescence is the number of contained arcs.

► **Theorem 2.** *There is an algorithm for UNWEIGHTED DIRECTED STEINER TREE, which given an unweighted directed graph  $G = (V, A)$ , terminal set  $R \subseteq V$ , root  $r \in R$ ,  $\varepsilon > 0$ , and integer  $p$ , computes a  $(1 + \varepsilon)$ -approximation to the cheapest Steiner arborescence  $T \subseteq G$  with  $p \geq |V(T) \setminus R|$  in time  $2^{p^2/\varepsilon} \cdot n^{O(1)}$ .<sup>1</sup>*

<sup>1</sup> If the input to this optimization problem is malformed (e.g., if  $p$  is smaller than the number of Steiner vertices of any feasible solution) then the output of the algorithm can be arbitrary (cf. [28])

Can our techniques be utilized for the even more general case when arcs have weights? Interestingly, in contrast to the above theorem we can show that in general the DIRECTED STEINER TREE problem most likely does not admit such approximation schemes, even when allowing “non-efficient” runtimes of the form  $f(p, \varepsilon) \cdot n^{g(\varepsilon)}$  for any computable functions  $f$  and  $g$ . This follows from the next theorem, since setting  $\varepsilon$  to any constant, the existence of such a  $(1 + \varepsilon)$ -approximation algorithm would imply  $W[1] = \text{FPT}$ .

► **Theorem 3.** *For any constant  $\alpha$ , it is  $W[1]$ -hard to compute an  $\alpha$ -approximation of the optimum Steiner arborescence  $T$  for DIRECTED STEINER TREE parameterized by  $|V(T) \setminus R|$ , if the input graph is arc-weighted.*

Other common variants of STEINER TREE include the PRIZE COLLECTING STEINER TREE and STEINER FOREST problems. The latter takes as input an edge-weighted undirected graph  $G = (V, E)$  and a list  $\{s_1, t_1\}, \dots, \{s_k, t_k\}$  of terminal pairs, i.e.,  $R = \{s_i, t_i \mid 1 \leq i \leq k\}$ . A *Steiner forest* is a forest  $F$  in  $G$  for which each  $\{s_i, t_i\}$  pair is in the same connected component, and the objective is to minimize the total weight of the forest  $F$ . For this variant it is not hard to see that parametrizing by  $p = |V(F) \setminus R|$  cannot yield any approximation scheme, as a simple reduction from STEINER TREE shows that the problem is APX-hard even if the input has no Steiner vertices (cf. [17]). For the PRIZE COLLECTING STEINER TREE problem, the input is again a terminal set in an edge-weighted graph, but the terminals have additional costs. A solution tree is allowed to leave out a terminal but has to pay its cost in return (cf. [35]). It is also not hard to see that this problem is APX-hard, even if there are no Steiner vertices at all. These simple results show that our techniques to obtain approximation schemes reach their limit quite soon: with the exception of UNWEIGHTED DIRECTED STEINER TREE, most common variants of STEINER TREE seem not to admit approximation schemes for our parameter  $p$ . We are however able to generalize our EPAS to STEINER FOREST if we combine  $p$  with the number  $c$  of connected components in the optimum solution. In fact, our main result of Theorem 1 is a corollary of the next theorem, using only the first part of the above mentioned reduction from STEINER TREE. Due to this, it is not possible to have a parameterized approximation scheme for the parameter  $c$  alone, as such an algorithm would imply a polynomial time approximation scheme for the APX-hard STEINER TREE problem. Hence the following result necessarily needs to combine the parameters  $p$  and  $c$ .

► **Theorem 4.** *There is an algorithm for STEINER FOREST, which given an edge-weighted undirected graph  $G = (V, E)$ , a list  $\{s_1, t_1\}, \dots, \{s_k, t_k\} \subseteq V$  of terminal pairs,  $\varepsilon > 0$ , and integers  $p, c$ , computes a  $(1 + \varepsilon)$ -approximation to the cheapest Steiner forest  $F \subseteq G$  with at most  $c$  connected components and  $p \geq |V(F) \setminus R|$  where  $R = \{s_i, t_i \mid 1 \leq i \leq k\}$ , in time  $(2c)^{O((p+c)^2/\varepsilon^4)} \cdot n^{O(1)}$ .*

A topic tightly connected to parameterized algorithms is kernelization. We here use the framework of Lokshtanov et al. [28], who also give a thorough introduction to the topic. Loosely speaking, a *kernelization algorithm* runs in polynomial time, and, given an instance of a parameterized problem, computes another instance of the same problem, such that the size of the latter instance is at most  $f(p)$  for some computable function  $f$  in the parameter  $p$  of the input instance. The computed instance is called the *kernel*, and for an optimization problem it must be possible to efficiently convert an optimum solution to the kernel into an optimum solution to the input instance.

A fundamental result of parameterized complexity says that a problem is FPT if and only if it has a kernelization algorithm [12]. This means that for our parameter  $p$ , most likely

STEINER TREE does not have a kernelization algorithm, as it is  $W[2]$ -hard. For this reason, the focus of kernelization results have previously again shifted to special cases. By a folklore result, STEINER TREE is FPT for our parameter  $p$  if the input graph is planar (cf. [24]). Of particular interest are *polynomial kernels*, which have size polynomial in the input parameter. The idea is that computing the kernel in this case is an efficient preprocessing procedure for the problem, such that exhaustive search algorithms can be used on the kernel. Suchý [33] proved that UNWEIGHTED STEINER TREE parameterized by  $p$  admits a polynomial kernel if the input graph is planar.

Our aspirations again are to obtain results for inputs that are as general as possible, i.e., on unrestricted edge-weighted input graphs. We prove that STEINER TREE has a polynomial *lossy* (approximate) kernel, despite the fact that the problem is  $W[2]$ -hard: an  $\alpha$ -*approximate kernelization algorithm* is a kernelization algorithm that computes a new instance for which a given  $\beta$ -approximation can be converted into an  $\alpha\beta$ -approximation for the input instance in polynomial time. The new instance is now called a (*polynomial*) *approximate kernel*, and its size is again bounded as a function (a polynomial) of the parameter of the input instance.

Just as for our parameterized approximation schemes in Theorems 1 and 4, we prove the existence of a lossy kernel for STEINER TREE by a generalization to STEINER FOREST where we combine the parameter  $p$  with the number  $c$  of connected components in the optimum solution. Also, our lossy kernel can approximate the optimum arbitrarily well: we prove that for our parameter the STEINER FOREST problem admits a *polynomial size approximate kernelization scheme* (PSAKS), i.e., for every  $\varepsilon > 0$  there is a  $(1 + \varepsilon)$ -approximate kernelization algorithm that computes a polynomial approximate kernel. An easy corollary then is that STEINER TREE parameterized only by  $p$  also has a PSAKS, by setting  $c = 1$  in Theorem 5 and using the reduction from STEINER TREE to STEINER FOREST as above.

► **Theorem 5.** *There is a  $(1 + \varepsilon)$ -approximate kernelization algorithm for STEINER FOREST, which given an edge-weighted undirected graph  $G = (V, E)$ , a list  $\{s_1, t_1\}, \dots, \{s_k, t_k\} \subseteq V$  of terminal pairs, and integers  $p, c$ , computes an approximate kernel of size  $((p + c)/\varepsilon)^{2^{O(1/\varepsilon)}}$ , if for the optimum Steiner forest  $F \subseteq G$ ,  $p \geq |V(F) \setminus R|$  where  $R = \{s_i, t_i \mid 1 \leq i \leq k\}$ , the number of connected components of  $F$  is at most  $c$ , and  $\varepsilon > 0$ .<sup>1</sup>*

Analogous to approximation schemes, it is possible to distinguish between efficient and non-efficient kernelization schemes: a PSAKS is *size efficient* if the size of the approximate kernel is bounded by  $f(\varepsilon) \cdot p^{O(1)}$ , where  $p$  is the parameter and  $f$  is a computable function independent of  $p$ . Our bound on the approximate kernel size in Theorem 5 implies that we do *not* obtain a size efficient PSAKS for either STEINER FOREST or STEINER TREE. This is in contrast to the existence of efficient approximation schemes for the same parameters in Theorems 1 and 4. We leave open whether a size efficient PSAKS can be found in either case. Interestingly, we also do not obtain any PSAKS for the UNWEIGHTED DIRECTED STEINER TREE problem, even though by Theorem 2 an EPAS exists. We leave open whether a PSAKS can be found for this variant as well.

**Used techniques.** Our algorithms are based on the intuition that a Steiner tree containing only few Steiner vertices but many terminals must either contain a large component induced by terminals, or a Steiner vertex with many terminal neighbors forming a large star. A high-level description of our algorithms for UNWEIGHTED DIRECTED STEINER TREE and STEINER FOREST therefore is as follows. In each step a tree is found in the graph in polynomial time, which connects some terminals using few Steiner vertices. We save this tree as part of the approximate solution and then contract it in the graph. The vertex resulting from

the contraction is declared a terminal and the process repeats for the new graph. Previous results [24, 33] have also built on this straightforward procedure in order to obtain FPT algorithms and polynomial kernels for special cases of UNWEIGHTED DIRECTED STEINER TREE and UNWEIGHTED STEINER TREE. In particular, in the unweighted undirected setting it is a well-known fact (cf. [33]) that contracting an adjacent pair of terminals is always a safe option, as there always exists an optimum Steiner tree containing this edge. However this immediately breaks down if the input graph is edge-weighted, as an edge between terminals might be very costly and should therefore not be contained in any (approximate) solution.

Instead we employ more subtle contraction rules, which use the following intuition. Every time we contract a tree with  $\ell$  terminals we decrease the number of terminals by  $\ell - 1$  (as the vertex arising from a contraction is a terminal). Our ultimate goal would be to reduce the number of terminals to one—at this point, the edges that we contracted during the whole run connect all the terminals. Decreasing the number of terminals by one can therefore be seen as a “unit of work”. We will pick a tree with the lowest cost per unit of work done, and prove that as long as there are sufficiently many terminals left in the graph, these contractions only lose an  $\varepsilon$ -factor compared to the optimum. As soon as the number of terminals falls below a certain threshold depending on the given parameter, we can use an FPT algorithm computing the optimum solution in the remaining graph. This algorithm is parametrized by the number of terminals, which now is bounded by our parameter. For the variants of STEINER TREE considered in our positive results, such FPT algorithms can easily be obtained from the ones for STEINER TREE [16, 2]. Adding this exact solution to the previously contracted trees gives a feasible solution that is a  $(1 + \varepsilon)$ -approximation.

Each step in which a tree is contracted in the graph, can be seen as a *reduction rule* as used for kernelization algorithms. Typically, a proof for a kernelization algorithm will define a set of reduction rules and then show that the instance resulting from applying the rules exhaustively has size bounded as a function in the parameter. To obtain an  $\alpha$ -approximate kernelization algorithm, additionally it is shown that each reduction rule is  $\alpha$ -safe. Roughly speaking, this means that at most a factor of  $\alpha$  is lost when applying any number of  $\alpha$ -safe reduction rules.

Contracting edges in a directed graph may introduce new paths, which did not exist before. Therefore, for the UNWEIGHTED DIRECTED STEINER TREE problem, we need to carefully choose the arborescence to contract. In order to prove Theorem 2 we show that each contraction is a  $(1 + \varepsilon)$ -safe reduction rule. However, the total size of the graph resulting from exhaustively applying the contractions is not necessarily bounded as a function of our parameter. Thus we do not obtain an approximate kernel.

For STEINER FOREST the situation is in a sense the opposite. Choosing a tree to contract follows a fairly simple rule. On the downside however, the contractions we perform are not necessarily  $(1 + \varepsilon)$ -safe reduction rules. In fact there are examples in which a single contraction will lose a large factor compared to the optimum cost. We are still able to show however, that after performing all contractions exhaustively, any  $\beta$ -approximation to the resulting instance can be converted into a  $(1 + \varepsilon)\beta$ -approximation to the original input instance. Even though the total size of the resulting instance again cannot be bounded in terms of our parameter, for STEINER FOREST we can go on to obtain a PSAKS. For this we utilize a result of Lokshantov et al. [28], which shows how to obtain a PSAKS for STEINER TREE if the parameter is the number of terminals. This result can be extended to STEINER FOREST, and since our instance has a number of terminals bounded in our parameter after applying all contractions, we obtain Theorem 5.

Finally, to obtain our inapproximability result of Theorem 3, we use a reduction from the DOMINATING SET problem. It was recently shown by Chen and Lin [8] that this problem does not admit parameterized  $\alpha$ -approximation algorithms for any constant  $\alpha$ , if the parameter is the solution size, unless  $W[1] = FPT$ . We are able to exploit this to also show that no such algorithm exists for DIRECTED STEINER TREE with edge weights, under the same assumption.

All missing proofs of this paper are deferred to the full version of the paper [17].

**Related work.** As the STEINER TREE problem and its variants have been studied since decades, the literature on this topic is huge. We only present a selection of related work here, that was not yet mentioned above.

For planar graphs [4] it was shown that an EPTAS exists for STEINER TREE. For STEINER FOREST a 2-approximation can be computed in polynomial time on general inputs [1], but an EPTAS also exists if the input is planar [19]. If the UNWEIGHTED STEINER TREE problem is parametrized by the solution size, it is known [14] that no polynomial (exact) kernel exists, unless  $NP \subseteq coNP/poly$ . If the input is restricted to planar or bounded-genus graphs it was shown that polynomial kernels do exist for this parametrization [31]. It was later shown [33] that for planar graphs this is even true for our parameter  $p$ . For the DIRECTED STEINER TREE problem it is a long standing open problem whether a polylogarithmic approximation can be computed in polynomial time. It is known that an  $O(|R|^\varepsilon)$ -approximation can be computed in polynomial time [7], and an  $O(\log^2 n)$ -approximation in quasi-polynomial time [7]. A recent result [21] considers generalizations of DIRECTED STEINER TREE and characterizes which of these problems are FPT and which are  $W[1]$ -hard for parameter  $|R|$ .

## 2 The weighted undirected Steiner forest and Steiner tree problems

In this section we describe an approximate polynomial time preprocessing algorithm that returns an instance of STEINER FOREST containing at most  $O((p+c)^2/\varepsilon^4)$  terminals if there is a Steiner forest with at most  $p$  Steiner vertices and at most  $c$  connected components. We can use this algorithm in two ways. Either we can proceed with a kernelization derived from Lokshtanov et al. [28] and obtain a polynomial size lossy kernel (Theorem 5), or we can run an exact FPT algorithm derived from Dreyfus and Wagner [16] on the reduced instance, obtaining an EPAS running in single exponential time with respect to the parameters (Theorems 1 and 4). In both cases we use the combined parameter  $(p, c)$ .

We first rescale all weights so that every edge has weight strictly greater than 1. Then, in each step of our algorithm we pick a star, add it to the solution, and contract the star in the current graph. We repeat this procedure until the number of terminals falls below a specified bound depending on  $\varepsilon$ ,  $p$ , and  $c$ . To describe how we pick the star to be contracted in each step, we need to introduce the *ratio* of a star. Let  $C$  be a set of edges of a star, i.e., all edges of  $C$  are incident to a common vertex which is the *center* of the star, and denote by  $Q$  the set of terminals incident to  $C$ . Provided  $|Q| \geq 2$ , we define the *ratio* of  $C$  as  $w(C)/(|Q| - 1)$ , where  $w(C) = \sum_{e \in C} w(e)$ . Note that we allow  $C$  to contain only a single edge if it joins two terminals. Observe also that due to rescaling of edge weights each star has ratio strictly greater than 1.

In every step, our algorithm contracts a star with the best available ratio (i.e., the lowest ratio among all stars connecting at least two terminals). Due to the following lemma, a star with the best ratio has a simple form: it consists of the cheapest  $i$  edges incident to its center vertex and some terminal. As there are  $n$  possible center vertices and at most  $n$  incident edges to each center, the best ratio star can be found in time  $O(n^2)$ .

► **Lemma 6.** *Let  $v$  be a vertex and denote by  $q_1, q_2, \dots$  the terminals adjacent to  $v$ , where  $w(vq_1) \leq w(vq_2) \leq \dots$ , i.e., the terminals are ordered non-decreasingly by the weight of the corresponding edge  $vq_i$ . The star with the best ratio having  $v$  as its center has edge set  $\{vq_1, vq_2, \dots, vq_\ell\}$  for some  $\ell$ .*

To analyse our algorithm we need to keep track of the different graphs resulting from each contraction step  $t$ . Initially we set  $G_0$  to the input graph, and in each step  $t \geq 0$  we obtain a new graph  $G_{t+1}$  from  $G_t$  by contracting a set of edges  $C_t$  in  $G_t$ , such that  $C_t$  forms a star of minimum ratio in  $G_t$ . That is, we obtain  $G_{t+1}$  from  $G_t$  by identifying all vertices incident to edges in  $C_t$ , removing all resulting loops, and among the resulting parallel edges we delete all but the lightest one with respect to their weights. We also adjust the terminal pairs in a natural way: let  $v$  be the vertex of  $G_{t+1}$  resulting from contracting  $C_t$ . If  $G_t$  had a terminal pair  $\{s, t\}$  such that  $s$  is incident to some edge of  $C_t$  while  $t$  is not, then we introduce the terminal pair  $\{v, t\}$  for  $G_{t+1}$ . Also every terminal pair  $\{s, t\}$  of  $G_t$  for which neither  $s$  nor  $t$  is incident to any edge of  $C_t$  is introduced as a terminal pair of  $G_{t+1}$ . Any terminal pair for which both  $s$  and  $t$  are incident to edges of  $C_t$  is going to be connected by a path in the computed solution, as it will contain  $C_t$ . Hence, such a terminal pair can be safely removed.

The algorithm stops contracting best-ratio stars when there are less than  $\tau$  terminals left; the exact value of  $\tau$  depends on  $p, c$ , and the desired approximation factor and we specify it later. If this happens in step  $\hat{t}$ , the solution lifting algorithm takes a feasible solution  $F$  of  $G_{\hat{t}}$  and returns the union of  $F$  and  $\bigcup_{t=0}^{\hat{t}} C_t$ . Such a solution is clearly feasible, since we adapted the terminal pairs accordingly after each contraction.

For the purpose of analysis, we consider a solution in the current graph  $G_t$  that originates from a solution of the original instance  $G_0$ , but may contain edges that are heavier than those in  $G_t$ . More concretely, denote by  $F_0^*$  a solution in  $G_0$  with at most  $p$  Steiner vertices and at most  $c$  components, i.e.,  $F_0^*$  is a Steiner forest containing every  $s_i-t_i$  path. We remark that  $F_0^*$  may or may not be an optimum solution of  $G_0$ . Given  $F_t^*$  for  $t \geq 0$ , we modify this solution to obtain a new feasible solution  $F_{t+1}^*$  on the terminal pairs of  $G_{t+1}$ . Note that the edges of the contracted star  $C_t$  might not be part of  $F_t^*$ . We still mimic the contraction of the star in  $F_t^*$ : to obtain  $F_{t+1}^*$  from  $F_t^*$ , we identify all leaves of  $C_t$  (which are terminals by Lemma 6 and thus part of the solution  $F_t^*$ ) and possibly also the center  $v$  of  $C_t$  if it is in  $F_t^*$ . This results in a vertex  $v'$ . We now want to delete edges incident to  $v'$  in such a way that we are left with an acyclic feasible solution. If we delete an inclusion-wise minimal feedback edge set, we clearly get a feasible solution. Let  $Q_t$  denote the set of terminals incident to  $C_t$ . We choose a feedback edge set  $D_t$  for which every edge was incident to a vertex of  $Q_t$  before the contraction in  $F_t^*$ , i.e., an edge of  $G_t$  corresponding to an edge of  $D_t$  never connects two Steiner vertices. Note that such an inclusion-wise minimal feedback edge set always exists: if we delete all edges of  $F_t^*$  incident to  $Q_t$  except  $C_t$  and then contract  $C_t$ , we get an acyclic graph. See Figure 1 for an illustration.

The resulting graph is  $F_{t+1}^*$ , which now forms a forest connecting all terminal pairs of  $G_{t+1}$ . Note that for each edge in  $F_{t+1}^*$  there is a corresponding edge in  $G_{t+1}$ , which however may be lighter in  $G_{t+1}$ , as from each bundle of parallel edges in  $G_t$  we keep the lightest one, but this edge may not exist in  $F_t^*$ .

To show that our algorithm only loses an  $\varepsilon$ -factor compared to the cost of the solution  $F_0^*$ , we will compare the cost of the edges  $C_t$  contracted by our algorithm to the set  $D_t = E(F_{t+1}^*) \setminus E(F_t^*)$  of deleted edges of  $F_t^*$ . Note that there are at least  $|Q_t| - c$  edges in  $D_t$ , since we contracted  $Q_t$  terminals in the forest  $F_t^*$  with at most  $c$  connected components to obtain  $F_{t+1}^*$ , and a forest on  $n$  vertices and  $k$  components has  $n - k$  edges. We decrease the number of vertices of  $F_t^*$  by at least  $|Q_t| - 1$  (one more if the center of the star with edge



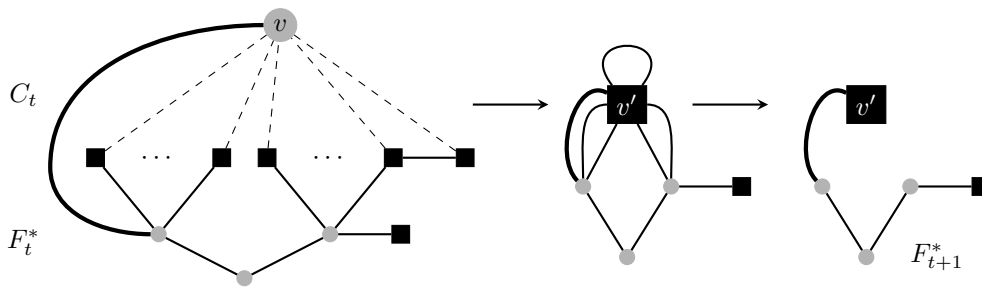


Figure 1 An example of creating  $F_{t+1}^*$  from  $F_t^*$  after a contraction  $C_t$ . Each edge in  $C_t$  (dashed) may or may not be in  $F_t^*$ . The thick edge cannot be in  $D_t$  because it is not incident to any terminal.

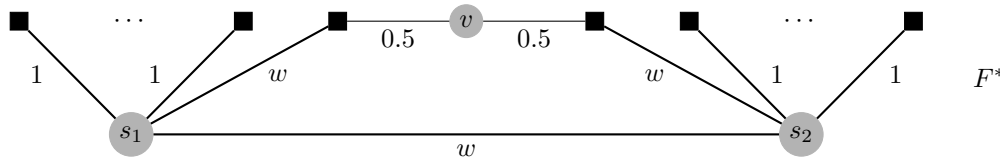


Figure 2 An example of a bad contraction. The numbers of terminals can be arbitrarily large and the weight  $w$  can be arbitrarily small. The star centered at  $v$  has ratio 1 while every star centered either at  $s_1$  or  $s_2$  has ratio slightly more than 1. By contracting the star centered at  $v$  we create a cycle containing only edges of weight  $w$ . Thus, for a sufficiently small value of  $w$  the contraction cannot be charged.

set  $C_t$  was a Steiner vertex present in  $F_t^*$ ), and we decrease the number of components by at most  $c - 1$ . Note also that for any two time steps  $t \neq t'$ , the sets  $D_t$  and  $D_{t'}$ , but also the sets  $C_t$  and  $C_{t'}$ , are disjoint. Thus if  $w(C_t) \leq (1 + \varepsilon)w(D_t)$  for every  $t$ , then our algorithm computes a  $(1 + \varepsilon)$ -approximation. Unfortunately, this is not always the case: there are contractions for which this condition does not hold (see Figure 2) and we have to account for them differently.

► **Definition 7.** If  $w(C_t) \leq (1 + \varepsilon)w(D_t)$  we say that the contracted edge set  $C_t$  in step  $t$  is *good*; otherwise  $C_t$  is *bad*. Moreover, if  $F_t^*$  has strictly more components than  $F_{t+1}^*$ , we say that  $C_t$  is *multiple-component*, otherwise it is *single-component*.

Our goal is to show that the total weight of bad contractions is bounded by an  $\varepsilon$ -fraction of the weight of  $F_0^*$ . We start by proving that if the set  $Q_t$  of terminals in  $C_t$  is sufficiently large, then the contraction is good. We define  $\lambda := (1 + \varepsilon)(p + c)/\varepsilon$ .

► **Lemma 8.** If  $|Q_t| \geq \lambda$ , then the contracted edge set  $C_t$  is good.

**Proof of Lemma 8.** For brevity, we drop the index  $t$ . Let  $r = w(C)/(|Q| - 1)$  be the ratio of the contracted star, and let  $\ell'$  be the number of deleted edges in  $D$  that connect two terminals. Note that any such edge has weight at least  $r$ , since it spans a star with two terminals, which has ratio equal to its weight, and since each edge in  $F^*$  (of which  $D$  is a subset) can only be heavier than the corresponding edge in the current graph  $G$ .

Let  $u_1, \dots, u_q$  be the Steiner vertices adjacent to edges in  $D$ , and let  $\ell_i$  be the number of edges in  $D$  incident to one such Steiner vertex  $u_i$ . Consider the star spanned by the  $\ell_i$  edges of  $D$  incident to  $u_i$ . If  $\ell_i \geq 2$ , the ratio of this star is at least  $r$ , since its edges are at least as heavy as the corresponding edges in  $G$  and the algorithm chose a star with the minimum ratio in  $G$ . Thus, the weight of edges in  $D$  incident to  $u_i$  is at least  $r(\ell_i - 1)$ . In the case where  $\ell_i = 1$ , the lower bound  $r(\ell_i - 1) = 0$  on the weight holds trivially.

Any edge in  $D$  not incident to any Steiner vertex  $u_i$  connects two terminals. Therefore, we have  $\ell' + \sum_{i=1}^q \ell_i = |D|$  as any edge in  $D$  is incident to a terminal in  $Q$  and we thus do not count any edge twice. Also recall that  $|D| \geq |Q| - c$ . Since  $F$  contains at most  $p$  Steiner vertices we have  $q \leq p$ , and we obtain

$$w(D) \geq r\ell' + \sum_{i=1}^q r(\ell_i - 1) = r \left( \ell' + \sum_{i=1}^q \ell_i - q \right) \geq r(|Q| - p - c).$$

Finally, using  $|Q| \geq \lambda$  we bound  $w(C)$  by  $(1 + \varepsilon)w(D)$  as follows:

$$\begin{aligned} (1 + \varepsilon)w(D) &\geq (1 + \varepsilon)r(|Q| - p - c) = r(|Q| - 1) + r(\varepsilon|Q| - (1 + \varepsilon)(p + c) + 1) \\ &\geq w(C) + r \left( \varepsilon \frac{(1 + \varepsilon)(p + c)}{\varepsilon} - (1 + \varepsilon)(p + c) \right) = w(C). \quad \blacktriangleleft \end{aligned}$$

Note that there may be a lot of contractions with  $|Q| < \lambda$ . However, we show that only a bounded number of them is actually bad. The key idea is to consider contractions with ratio in an interval  $((1 + \delta)^i; (1 + \delta)^{i+1}]$  for some  $\delta > 0$  and integer  $i$ . Due to the rescaling of weights every star belongs to an interval with  $i \geq 0$ . The following crucial lemma of our analysis shows that the number of bad single-component contractions in each such interval is bounded in terms of  $p$  and  $\varepsilon$ , if  $\delta$  is a function of  $\varepsilon$ . In particular, let  $\delta := \sqrt{1 + \varepsilon} - 1$ , so that  $(1 + \delta)^2 = 1 + \varepsilon$ . We call an edge set  $C$  with ratio  $r$  in the  $i$ -th interval, i.e., with  $r \in ((1 + \delta)^i; (1 + \delta)^{i+1}]$ , an  $i$ -contraction, and define  $\kappa := (1 + \delta)p/\delta$ .

► **Lemma 9.** *For any  $i$  the number of bad single-component  $i$ -contractions is at most  $\kappa$ .*

**Proof.** Let us focus on bad single-component  $i$ -contractions only which we just call bad  $i$ -contractions for brevity. Suppose for a contradiction that the number of bad  $i$ -contractions is larger than  $\kappa$ . Let  $\tilde{t}$  be the first step with a bad  $i$ -contraction, i.e.,  $\tilde{t}$  is the minimum among all  $t$  for which  $w(C_t) > (1 + \varepsilon)w(D_t)$  and  $w(C_t)/(|Q_t| - 1) \in ((1 + \delta)^i; (1 + \delta)^{i+1}]$ . The plan is to show that at step  $\tilde{t}$  there is a “light” star in  $G_{\tilde{t}}$  with ratio at most  $(1 + \delta)^i$  and consequently the algorithm would do a  $j$ -contraction for some  $j < i$ . This leads to a contradiction, since we assumed that in step  $\tilde{t}$  the contraction has ratio in interval  $i$ . Note that it is sufficient to find such a light star in  $F_{\tilde{t}}^*$  as for each edge in  $F_{\tilde{t}}^*$  there is an edge in the graph  $G_{\tilde{t}}$  between the same vertices of the same weight or even lighter.

We claim that for each step  $t$  in which the algorithm does a bad  $i$ -contraction there is an edge  $e_t \in D_t$  with weight at most  $(1 + \delta)^{i-1}$ . We have  $w(C_t) > (1 + \varepsilon)w(D_t)$  as  $C_t$  is bad and  $w(C_t) \leq (1 + \delta)^{i+1}(|Q_t| - 1)$  as the ratio of  $C_t$  is in interval  $i$ . Putting it together and using the definition of  $\delta$  we obtain  $w(D_t) < (1 + \delta)^{i+1}/(1 + \varepsilon) \cdot (|Q_t| - 1) = (1 + \delta)^{i-1}(|Q_t| - 1)$ . Because  $C_t$  is single-component, we have  $|D_t| \geq |Q_t| - 1$  and therefore there is an edge  $e_t \in D_t$  with weight at most  $(1 + \delta)^{i-1}$ , which proves the claim.

Since edges between terminals have weight at least  $(1 + \delta)^i$  in step  $t$  (recall that each such edge is a star with ratio equal to its weight), the edge  $e_t$  is incident to a Steiner vertex in  $F_t^*$  (otherwise the algorithm would have chosen one of the edges between terminals to contract). As  $D_t \cap D_{t'} = \emptyset$ , the edges  $e_t$  and  $e_{t'}$  for steps  $t \neq t'$  with bad  $i$ -contractions are distinct. Let  $S$  be the set of such light edges  $e_t$ . We have  $|S| > \kappa$  as there are more than  $\kappa$  bad  $i$ -contractions.

Since the solution  $F_0^*$  uses at most  $p$  Steiner vertices, also  $F_{\tilde{t}}^*$  contains at most  $p$  of them. Therefore at  $\tilde{t}$  there must be a Steiner vertex  $v$  in  $F_{\tilde{t}}^*$  incident to at least  $|S|/p > \kappa/p \geq (1 + \delta)/\delta$  edges in  $S$ . Consider a star  $C$  with  $v$  as the center and with edges from  $S$  that are incident to  $v$ ; we have  $|C| \geq (1 + \delta)/\delta$ . The ratio of this star is at most  $|C|(1 + \delta)^{i-1}/(|C| - 1)$ .

Since  $|C|/(|C| - 1) \leq (1 + \delta)$  (by a routine calculation) we get that the ratio of  $C$  is at most  $(1 + \delta)^i$  which is a contradiction to the assumption that the algorithm does an  $i$ -contraction in step  $\tilde{t}$ . ◀

We also need a bound on number of bad multiple-component edge sets.

► **Lemma 10.** *The number of steps  $t$  in which a bad multiple-component edge set  $C_t$  is contracted is at most  $c - 1$ .*

**Proof.** If  $C_t$  is a bad multiple-component edge set,  $F_{t+1}^*$  must have at least one component fewer than  $F_t^*$ . Since  $F_0^*$  has at most  $c$  components, the bound follows. ◀

We remark that the proofs of Lemmas 9 and 10 do not use that the number of terminals in a bad  $i$ -contraction is bounded by  $\lambda$ , as shown in Lemma 8. Instead we bound the total weight of bad contractions in terms of  $\lambda$ . For this let  $j$  be the largest interval of any contraction during the whole run of the algorithm, i.e., the ratio of every contracted star is at most  $(1 + \delta)^{j+1}$ . As there are at most  $\kappa$  bad single-component contractions in each interval and  $c$  bad multiple-component contractions and the interval size grows exponentially, we can upper bound the total weight of bad contractions in terms of  $\kappa, c, \lambda$  and  $(1 + \delta)^j$ . We can also lower bound the weight of  $w(F_0^*)$  in terms of  $(1 + \delta)^j$  and the lower bound  $\tau$  on the number of terminals in the graph. If  $\tau$  is large enough then the total weight of edge sets  $C_t$  of bad contractions is at most  $\varepsilon \cdot w(F_0^*)$ . These ideas are summarized in the next lemma.

► **Lemma 11.** *Let  $j$  be the largest interval of any contraction during the whole run of the algorithm and let  $W_B$  be the total weight of edge sets  $C_t$  of bad contractions. Then, the following holds.*

1.  $W_B \leq (\kappa + c) \cdot \lambda \cdot (1 + \delta)^{j+2} / \delta$ .
2.  $w(F_0^*) \geq (1 + \delta)^j \cdot (\tau - 2p - c)$ .
3. If  $\tau := (\kappa + c) \cdot \lambda \cdot (1 + \delta)^2 / (\varepsilon \delta) + 2p + c$  then  $W_B \leq \varepsilon \cdot w(F_0^*)$ .

The above lemma can now be used to prove that all the contractions put together (by scaling  $\varepsilon$ ) form a  $(1 + \varepsilon)$ -approximate pre-processing procedure with respect to  $F_0^*$ .

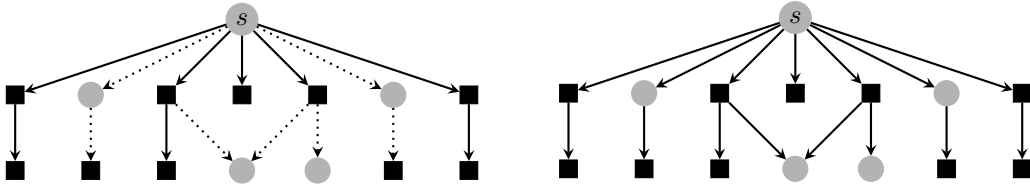
► **Lemma 12.** *The algorithm outputs an instance with  $\tau \in O((p + c)^2 / \varepsilon^4)$  terminals and (together with the solution lifting algorithm) it is a  $(1 + 2\varepsilon)$ -approximate polynomial time pre-processing algorithm with respect to  $F_0^*$ .*

Note that in case the given  $p$  is smaller than the number of Steiner vertices in  $F_0^*$ , or  $c$  is smaller than the number of connected components in  $F_0^*$ , the algorithm still outputs a Steiner forest, but the approximation factor may be arbitrary. The last lemma can easily be used to prove Theorems 1, 4 and 5.

### 3 The unweighted directed Steiner tree problem

In this section we provide an EPAS for the UNWEIGHTED DIRECTED STEINER TREE problem, in which each arc has unit weight. The idea behind our algorithm given in this section is to reduce the number of terminals of the input instance via a set of reduction rules. That is, we would like to reduce the input graph  $G$  to a graph  $H$ , and prove that the number of terminals in  $H$  is bounded by a function of our parameter  $p$  and the desired approximation ratio. On  $H$  we use the algorithm of Björklund et al. [2] to obtain an optimum solution.

Our first reduction rule represents the idea that a terminal in the immediate neighborhood of the root can be contracted to the root. Observe that in this case our algorithm has to



■ **Figure 3** An example of extended neighborhood of Steiner vertex  $s$ . The set  $N_{\text{Ext}}^0(s)$  is depicted on the left using full arcs, while the vertices connected by dotted arcs are not a part of this set. The set  $N_{\text{Ext}}^1(s)$  is depicted on the right using full arcs.

pay 1 for connecting such a terminal to the root, however, any feasible solution must connect this terminal as well using at least one arc—this argument is formalized in Lemma 13.

► **Reduction Rule R1.** *If there is an arc from the root  $r$  to a terminal  $v \in R$ , we contract the arc  $(r, v)$ , and declare the resulting vertex the new root.*

► **Lemma 13.** *Reduction Rule R1 is 1-safe and can be implemented in polynomial time. Furthermore, there is a solution lifting algorithm running in polynomial time and returning a Steiner arborescence if it gets a Steiner arborescence of the reduced graph as input.*

The idea behind our next reduction rule is the following. Assume there is a Steiner vertex  $s$  in the optimum arborescence  $T$  connected to many terminals with paths not containing any other Steiner vertices. We can then afford to buy all these paths emanating from  $s$  together with a path connecting the root to  $s$ . Formally, we say that a vertex  $u$  is a  $k$ -extended neighbor of some vertex  $v$ , if there exists a directed path  $P$  starting in  $v$  and ending in  $u$ , such that  $V(P) \setminus \{v\}$  contains at most  $k$  Steiner vertices. Note that a vertex is always a  $k$ -extended neighbor of itself for any  $k$ , and that each of the above terminals connected to  $s$  in  $T$  is a 0-extended neighbor of  $s$ . We denote by  $N_{\text{Ext}}^k(v)$  the set of all  $k$ -extended neighbors of  $v$ , and call it the  $k$ -extended neighborhood of  $v$  (see Figure 3). By the following observation the Steiner vertex  $s$  of  $T$  lies in the  $p$ -extended neighborhood of the root  $r$ . Therefore there is a path containing at most  $p$  Steiner vertices connecting  $r$  to  $s$ .

► **Observation 14.** *Let  $G = (V, A)$  be a directed graph with root  $r \in R$ . Suppose there exists a Steiner arborescence  $T \subseteq G$  with at most  $p$  Steiner vertices. It follows that  $V(T) \subseteq N_{\text{Ext}}^p(r)$ .*

In what follows we fix  $\varepsilon > 0$ . The second reduction rule contracts a path from  $r$  to a Steiner vertex  $s$  in the  $p$ -extended neighborhood of  $r$  together with the 0-extended neighborhood of  $s$  if this neighborhood is sufficiently large.

► **Reduction Rule R2.** *If there exists a Steiner vertex  $s$  with  $|N_{\text{Ext}}^0(s)| \geq p/\varepsilon$  and  $s \in N_{\text{Ext}}^p(r)$ , so that there is an  $r \rightarrow s$  path  $P$  containing at most  $p$  Steiner vertices, then we contract the subgraph of  $G$  induced by  $N_{\text{Ext}}^0(s)$  and  $P$  in  $G$ , and declare the resulting vertex the new root.*

► **Lemma 15.** *Reduction Rule R2 is  $(1 + \varepsilon)$ -safe and can be implemented in polynomial time. Furthermore, there is a solution lifting algorithm running in polynomial time and returning a Steiner arborescence if it gets a Steiner arborescence of the reduced graph as input.*

Now we prove that if none of the above reduction rules is applicable and our algorithm was provided with a correct value for parameter  $p$ , then the number of terminals in the reduced graph can be bounded by  $p^2/\varepsilon$ .

► **Lemma 16.** *Let  $G$  be an instance of DIRECTED STEINER TREE, and denote by  $H$  the graph obtained from  $G$  by exhaustive application of Reduction Rules R1 and R2. Suppose*

that there exists a Steiner arborescence in  $G$  containing at most  $p$  Steiner vertices. It follows that the remaining terminal set  $R$  of  $H$  has size less than  $p^2/\varepsilon$ .

The last step of the algorithm is to compute an optimum solution in the graph  $H$  obtained from the input graph  $G$  after exhaustively applying the two above reduction rules. From the resulting arborescence in  $H$  we obtain an arborescence in  $G$  by running the solution lifting algorithms for each reduction rule applied (in the reverse order); the existence and correctness of the solution lifting algorithms for our reduction rules is provided by Lemmas 13 and 15.

---

## References

- 1 Ajit Agrawal, Philip Klein, and R. Ravi. When Trees Collide: An Approximation Algorithm for the Generalized Steiner Problem on Networks. *SIAM Journal on Computing*, 24(3):440–456, jun 1995. doi:10.1137/s0097539792236237.
- 2 Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Fourier meets möbius: fast subset convolution. In David S. Johnson and Uriel Feige, editors, *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 67–74. ACM, 2007. doi:10.1145/1250790.1250801.
- 3 Edouard Bonnet, Bruno Escoffier, Eun Jung Kim, and Vangelis Th. Paschos. On subexponential and fpt-time inapproximability. In Gregory Gutin and Stefan Szeider, editors, *Parameterized and Exact Computation - 8th International Symposium, IPEC 2013, Sophia Antipolis, France, September 4-6, 2013, Revised Selected Papers*, volume 8246 of *Lecture Notes in Computer Science*, pages 54–65. Springer, 2013. doi:10.1007/978-3-319-03898-8\_6.
- 4 Glencora Borradaile, Philip N. Klein, and Claire Mathieu. An  $O(n \log n)$  approximation scheme for steiner tree in planar graphs. *ACM Trans. Algorithms*, 5(3):31:1–31:31, 2009. doi:10.1145/1541885.1541892.
- 5 Jaroslaw Byrka, Fabrizio Grandoni, Thomas Rothvoß, and Laura Sanità. Steiner tree approximation via iterative randomized rounding. *J. ACM*, 60(1):6:1–6:33, 2013. doi:10.1145/2432622.2432628.
- 6 Parinya Chalermsook, Marek Cygan, Guy Kortsarz, Bundit Laekhanukit, Pasin Manurangsi, Danupon Nanongkai, and Luca Trevisan. From Gap-ETH to FPT-Inapproximability: Clique, Dominating Set, and More. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, oct 2017. doi:10.1109/focs.2017.74.
- 7 Moses Charikar, Chandra Chekuri, To-Yat Cheung, Zuo Dai, Ashish Goel, Sudipto Guha, and Ming Li. Approximation algorithms for directed steiner problems. *J. Algorithms*, 33(1):73–91, 1999. doi:10.1006/jagm.1999.1042.
- 8 Yijia Chen and Bingkai Lin. The Constant Inapproximability of the Parameterized Dominating Set Problem. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, oct 2016. doi:10.1109/focs.2016.61.
- 9 Rajesh Chitnis, Andreas Emil Feldmann, and Pasin Manurangsi. Parameterized Approximation Algorithms for Directed Steiner Network Problems. *arXiv preprint*, 2017. arXiv:1707.06499.
- 10 Rajesh Hemant Chitnis, MohammadTaghi Hajiaghayi, and Guy Kortsarz. Fixed-parameter and approximation algorithms: A new look. In Gregory Gutin and Stefan Szeider, editors, *Parameterized and Exact Computation - 8th International Symposium, IPEC 2013, Sophia Antipolis, France, September 4-6, 2013, Revised Selected Papers*, volume 8246 of *Lecture Notes in Computer Science*, pages 110–122. Springer, 2013. doi:10.1007/978-3-319-03898-8\_11.
- 11 Miroslav Chlebík and Janka Chlebíková. Approximation hardness of the steiner tree problem on graphs. In Martti Penttonen and Erik Meineche Schmidt, editors, *Algorithm Theory*

- *SWAT 2002, 8th Scandinavian Workshop on Algorithm Theory, Turku, Finland, July 3-5, 2002 Proceedings*, volume 2368 of *Lecture Notes in Computer Science*, pages 170–179. Springer, 2002. doi:10.1007/3-540-45471-3\_18.
- 12 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshantov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.
  - 13 Irit Dinur and David Steurer. Analytical approach to parallel repetition. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 624–633. ACM, 2014. doi:10.1145/2591796.2591884.
  - 14 Michael Dom, Daniel Lokshantov, and Saket Saurabh. Kernelization lower bounds through colors and ids. *ACM Trans. Algorithms*, 11(2):13:1–13:20, 2014. doi:10.1145/2650261.
  - 15 Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, 1999. doi:10.1007/978-1-4612-0515-9.
  - 16 S. E. Dreyfus and R. A. Wagner. The steiner problem in graphs. *Networks*, 1(3):195–207, 1971. doi:10.1002/net.3230010302.
  - 17 Pavel Dvořák, Andreas Emil Feldmann, Dušan Knop, Tomáš Masařík, Tomáš Toufar, and Pavel Veselý. Parameterized Approximation Schemes for Steiner Trees with Small Number of Steiner Vertices. *arXiv preprint*, 2017. arXiv:1710.00668v2.
  - 18 Eduard Eiben, Mithilesh Kumar, Amer E Mouawad, and Fahad Panolan. Lossy kernels for connected dominating set on sparse graphs. *arXiv preprint*, 2017. arXiv:1706.09339.
  - 19 David Eisenstat, Philip Klein, and Claire Mathieu. An efficient polynomial-time approximation scheme for Steiner forest in planar graphs. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 626–638. Society for Industrial and Applied Mathematics, jan 2012. doi:10.1137/1.9781611973099.53.
  - 20 Andreas Emil Feldmann. Fixed parameter approximations for k-center problems in low highway dimension graphs. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part II*, volume 9135 of *Lecture Notes in Computer Science*, pages 588–600. Springer, 2015. doi:10.1007/978-3-662-47666-6\_47.
  - 21 Andreas Emil Feldmann and Dániel Marx. The complexity landscape of fixed-parameter directed steiner network problems. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, volume 55 of *LIPICs*, pages 27:1–27:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016. doi:10.4230/LIPICs.ICALP.2016.27.
  - 22 Eran Halperin and Robert Krauthgamer. Polylogarithmic inapproximability. In Lawrence L. Larmore and Michel X. Goemans, editors, *Proceedings of the 35th Annual ACM Symposium on Theory of Computing, June 9-11, 2003, San Diego, CA, USA*, pages 585–594. ACM, 2003. doi:10.1145/780542.780628.
  - 23 Frank K Hwang, Dana S Richards, and Pawel Winter. *The Steiner tree problem*, volume 53. Elsevier, 1992. doi:10.1016/s0167-5060(08)x7008-6.
  - 24 Mark Jones, Daniel Lokshantov, M. S. Ramanujan, Saket Saurabh, and Ondrej Suchý. Parameterized complexity of directed steiner tree on sparse graphs. In Hans L. Bodlaender and Giuseppe F. Italiano, editors, *Algorithms - ESA 2013 - 21st Annual European Symposium, Sophia Antipolis, France, September 2-4, 2013. Proceedings*, volume 8125 of *Lecture Notes in Computer Science*, pages 671–682. Springer, 2013. doi:10.1007/978-3-642-40450-4\_57.
  - 25 Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Plenum, 1972. doi:10.1007/978-1-4684-2001-2\_9.

- 26 R. Krithika, Pranabendu Misra, Ashutosh Rai, and Prafullkumar Tale. Lossy kernels for graph contraction problems. In Akash Lal, S. Akshay, Saket Saurabh, and Sandeep Sen, editors, *36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2016, December 13-15, 2016, Chennai, India*, volume 65 of *LIPIcs*, pages 23:1–23:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016. doi:10.4230/LIPIcs.FSTTCS.2016.23.
- 27 Michael Lampis. Parameterized approximation schemes using graph widths. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, volume 8572 of *Lecture Notes in Computer Science*, pages 775–786. Springer, 2014. doi:10.1007/978-3-662-43948-7\_64.
- 28 Daniel Lokshantov, Fahad Panolan, M. S. Ramanujan, and Saket Saurabh. Lossy kernelization. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 224–237. ACM, 2017. doi:10.1145/3055399.3055456.
- 29 Dániel Marx. Parameterized complexity and approximation algorithms. *Comput. J.*, 51(1):60–78, 2008. doi:10.1093/comjnl/bxm048.
- 30 Daniel Mölle, Stefan Richter, and Peter Rossmanith. Enumerate and expand: Improved algorithms for connected vertex cover and tree cover. *Theory Comput. Syst.*, 43(2):234–253, 2008. doi:10.1007/s00224-007-9089-3.
- 31 Marcin Pilipczuk, Michal Pilipczuk, Piotr Sankowski, and Erik Jan van Leeuwen. Network Sparsification for Steiner Problems on Planar and Bounded-Genus Graphs. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*. IEEE, oct 2014. doi:10.1109/focs.2014.37.
- 32 Sebastian Siebertz. Lossy kernels for connected distance- $r$  domination on nowhere dense graph classes. *arXiv preprint*, 2017. arXiv:1707.09819.
- 33 Ondrej Suchý. Extending the kernel for planar steiner tree to the number of steiner vertices. *Algorithmica*, 79(1):189–210, 2017. doi:10.1007/s00453-016-0249-1.
- 34 Andreas Wiese. A  $(1+\epsilon)$ -approximation for unsplittable flow on a path in fixed-parameter running time. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 67:1–67:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPIcs.ICALP.2017.67.
- 35 David P Williamson and David B Shmoys. *The design of approximation algorithms*. Cambridge university press, 2011. doi:10.1017/cbo9780511921735.